Neural Network Model Performance Report

Overview:

The purpose of this project was to analyze the given dataset from Alphabet Soup who wanted a tool that can help them select the applicants who had the best chances of success in their ventures. This is a deep dive into the more than 34,000 organizations that have received funding from Alphabet Soup over the years that captured metadata across all areas. After creating the model, I was tasked to achieve an accuracy higher than 75%.

Results:

I. Data Preprocessing
   a. The target variable for this project is 'IS_SUCCESSFUL'
   b. The feature variable for this project are all other variables contained in this dataset, excluding the 'EIN' and 'NAME' columns
   c. The 'EIN' and 'NAME' variables were removed because they were not our target or featured variables and would only confuse the neural networks performance.

II. Compiling, Training and Evaluation
   a. First Run
      i. The two hidden layers had an activation of 'RELU' while the output layer used 'SIGMOID' with 100 epochs
         1. Hidden layer one = 10
         2. Hidden layer two = 5
         3. Saved in AlphabetSoupCharity_1.h5

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 10)                720
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 55
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 6
=================================================================
Total params: 781
Trainable params: 781
Non-trainable params: 0
```

      ii.
      iii. After the first run the accuracy was 73.09%, slightly below what is required by Alphabet Soup. To increase performance, I shifted the neurons for better optimization
   b. Second run
      i. In this run there were three hidden layers that had an activation of 'RELU' while the output layer used 'SIGMOID' with 100 epochs
         1. Hidden layer one = 7
         2. Hidden layer two = 14

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 7)                 504
_________________________________________________________________
dense_1 (Dense)              (None, 14)                112
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 15
=================================================================
Total params: 631
Trainable params: 631
Non-trainable params: 0
```

ii. After this second run the accuracy was 73.03%, still not where it needed to be for Alphabet Soup.

c. Third run

i. With two unsuccessful attempts to increase performance of the learning model, I went back through my dataset to see if there could be a problem with the data being filtered in.

1. Increased the cutoff values for application types and classifications.
2. Run multiple attempts

ii. Attempt One

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 8)                 400
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 45
_________________________________________________________________
dense_2 (Dense)              (None, 5)                 30
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 6
=================================================================
Total params: 481
Trainable params: 481
Non-trainable params: 0
```

1.
2. Layer one = 8
3. Layer two = 5
4. Layer three = 5
5. Results: 73.30%

iii. Attempt Two

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_4 (Dense)              (None, 16)                800
_________________________________________________________________
dense_5 (Dense)              (None, 10)                170
_________________________________________________________________
dense_6 (Dense)              (None, 10)                110
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 11
=================================================================
Total params: 1,091
Trainable params: 1,091
Non-trainable params: 0
```

1.
2. Layer one = 16
3. Layer two = 10
4. Layer three = 10

5. Results: 73.30

iv. Attempt Three

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_8 (Dense)              (None, 32)                1600
_________________________________________________________________
dense_9 (Dense)              (None, 10)                330
_________________________________________________________________
dense_10 (Dense)             (None, 10)                110
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 11
=================================================================
Total params: 2,051
Trainable params: 2,051
Non-trainable params: 0
```

1.
2. Layer one = 32
3. Layer two = 10
4. Layer three = 10

Summary:

Overall, I was unable to increase the performance of the deep learning model to fit the parameters of Alphabet Soup for success prediction. I tried many ways to do so by changing the number of neurons and hidden layers, playing around with multiple attempts to train. In the last optimization I thought that by increasing the neurons and running the model test it would help in some way but that was not the case.